The effect of the fragmentation problem in decision tree learning applied to the search for single top quark production

To cite this article: R Vilalta *et al* 2010 *J. Phys.: Conf. Ser.* **219** 032063

The effect of the fragmentation problem in decision tree learning applied to the search for single top quark production

R Vilalta[1], R Valerio[2], F Ocegueda-Hernandez[1], and G Watts[3]

[1] Department of Computer Science, University of Houston, 4800 Calhoun Rd., Houston TX 77204-3010, USA
[2] Center for Research and Advanced Studies, CINVESTAV, Avenida Científica No. 1145, Zapopan Jalisco, 45015, México
[3] Department of Physics, University of Washington, Box 351560, Seattle WA 98195-1560, USA

E-mail: `vilalta@cs.uh.edu, rvalerio@gdl.cinvestav.mx`

Abstract. Decision tree learning constitutes a suitable approach to classification due to its ability to partition the variable space into regions of class-uniform events, while providing a structure amenable to interpretation, in contrast to other methods such as neural networks. But an inherent limitation of decision tree learning is the progressive lessening of the statistical support of the final classifier as clusters of single-class events are split on every partition, a problem known as the fragmentation problem. We describe a software system called DTFE, for Decision Tree Fragmentation Evaluator, that measures the degree of fragmentation caused by a decision tree learner on every event cluster. Clusters are found through a decomposition of the data using a technique known as Spectral Clustering. Each cluster is analyzed in terms of the number and type of partitions induced by the decision tree. Our domain of application lies on the search for single top quark production, a challenging problem due to large and similar backgrounds, low energetic signals, and low number of jets. The output of the machine-learning software tool consists of a series of statistics describing the degree of data fragmentation.

1. Introduction

Decision tree learning algorithms stand as a popular non-parametric approach to classification; the general idea is to have the input or variable space iteratively partitioned into smaller regions until each region exhibits an approximately uniform class distribution. Decision tree learning algorithms have gained increasing acceptance in the physics community because of several factors: the user is relieved from establishing parametric model assumptions; the output is amenable to interpretation; accuracy performance tends to be competitive when compared to other techniques, such as neural networks and support vector machines; and CPU cost during training is relatively low. Nevertheless, an inherent limitation of top-down decision tree algorithms exists, also known as the *fragmentation problem*, in which the continuous partitioning of the training set at every tree node reduces the number of examples or events, the statistical support, at lower-level nodes [1]. An alternative view of the fragmentation problem is as follows. If the signal class can be modeled as a mixture of probability model components, such as a mixture of Gaussians, each attempt by the decision tree to separate one model component from the opposite class may end up producing decision boundaries that cut across other signal components. Too much partitioning ends up splitting signal components in excess, weakening their statistical support. Under such scenario it is common to experience multiple misclassifications [1–7].

In this paper we describe a system, called DTFE, for Decision-Tree Fragmentation Evaluator, to measure the degree of fragmentation produced by a decision tree. In essence we decompose both signal and background distributions into a set of clusters over the variable space. Each cluster is then analyzed in terms of the number and type of cuts imposed by the decision tree. We base our analysis on the statistical weight of each fragment, a value taken from the degree of data concentration inside the fragment. The motivation behind DTFE is to understand the reasons explaining the behavior of a decision tree classifier over different input-output distributions.

We show how the fragmentation problem is not necessarily always present in classification, or signal-background separation problems. The degree of presence of the fragmentation problem depends on the particular distribution of class-clusters over the variable space. Specifically, fragmentation occurs when a cluster splits into two or more sub-clusters due to the coarse mechanism of a decision tree algorithm that employs hyper-rectangles –simple decision boundaries– over the variable space.

Our experimental results are based on single top quark production Monte Carlo data. Both signal and background events are first decomposed into a mixture of clusters. Each cluster is then analyzed in terms of the number of times it is partitioned. Our software DTFE gathers statistics in a bottom-top fashion about each fragment to produce a set of class-average statistics.

This paper is organized as follows. Section 2 provides an overview of decision tree induction and defines the fragmentation problem; Section 3 describes our algorithm, DTFE, that measures the degree of fragmentation by a decision tree learner. Section 4 shows experimental results. Lastly, Section 6 gives a summary and conclusions.

2. Decision Trees and the Fragmentation Problem

The automatic separation of signals from background can be cast as a classification problem. We assume an input training set $S = \{(\mathbf{x}, y)\}$, where each event $\mathbf{x} = (a_1, a_2, \cdots, a_n)$ is a vector of features or variables, such as energy, momentum, invariant mass, etc. Vector $\mathbf{x}$ is a point in the input space $\mathcal{X}$, and we assign to it a class label y from the output space $\mathcal{Y} = \{0, 1\}$, with $y = 0$ representing background and $y = 1$ representing signal. The outcome of the classifier is a function f mapping the input space to the output space, $f : \mathcal{X} \rightarrow \mathcal{Y}$. Function f can then be used to predict the class of previously unseen events. Our main interest is in the ability of f to correctly *predict* the class of events outside S. We look for hypotheses not only consistent with S, but that *generalize* beyond that set.

A decision tree is a classifier that uses a divide-and-conquer strategy. Proceeding top-down, the root of the tree is formed by selecting a variable[1] A that splits the training set into mutually exclusive subsets $S_0, S_1, \cdots, S_m$, where each S_i contains all events with the same value for A. Commonly A is selected via some class-impurity measure, like entropy, gini, Laplace, and χ^2, which yields axis-parallel partitions over the input space. Other less common techniques look for combinations of features at each node that produce non-axis-parallel partitions.

The same methodology is recursively applied on each S_i to construct the child subtrees respectively. A subset S_i represents a leaf if most of its examples belong to the same class, or if S_i is too small (i.e., if $|S_i| < \epsilon$, where ϵ is user defined); the majority class in S_i is associated with that leaf. An example $\mathbf{x}$ is classified, starting from the root of the tree, by iteratively following the branch that matches the value of the node variable. At the end of the path, the class attached to that leaf is assigned to $\mathbf{x}$. An example of the output of a decision tree algorithm is shown in Fig. 1-right.

[1] We use upper case for a variable, such as A, and lower case for a variable value, such as a.

2.1. The Fragmentation Problem

We now explain the fragmentation problem in detail. A decision tree algorithm assumes a representation where each class is the disjunction of several sub-concepts, also known as a Disjunctive Normal Form or DNF representation. Each branch from the root of the tree to a terminal node or leaf stands as one sub-concept or disjunct. As an example assume a variable space made of two variables A_1 and A_2 and a distribution of events as shown in Fig. 1-left. A possible decision tree corresponding to the partitions shown over the two-dimensional variable space is shown in Fig. 1-right. In this example the signal class ($y = 1$) divides into two clusters but the initial attempt of the decision tree to isolate the upper left cluster from the background class results in cutting the second or lower left cluster in two parts.

Decision tree algorithms carry out a continuous partition-refinement over the variable space; every tree branch is grown until a terminal node or leaf delineates a single-class region. A limitation inherent to this approach is that, while searching for a region that is class uniform, each splitting of the training data may separate or pull apart examples in support of a different disjunct. This situation not only requires that several approximations be found on dispersed regions of the variable space, but also reduces the support or evidential credibility of each individual disjunct, eventually complicating its identification. This problem is known as the *fragmentation problem*. As an example, Fig. 2-left shows a configuration of events similar to Fig. 1 but where splitting the lower left cluster leaves few signal events in a region where the background class turns dominant; the signal class is so poorly represented in this region that no further attempt is made to separate signal from background. Fig. 2-right shows the corresponding tree where a fragmented portion of the lower signal cluster is now misclassified.

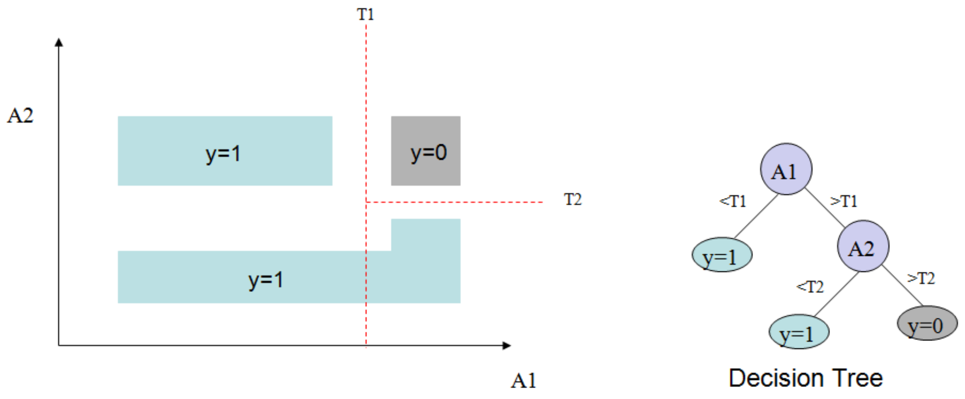

Figure 1. Left. A 2-dimensional variable space with 2 classes: signal or positive, $y = 1$, and background or negative $y = 0$. Right. The corresponding decision tree.

The fragmentation problem originates from two main reasons. First is the requirement that the region covered by each different disjunct, equivalently each path from the root to a leaf node, be mutually exclusive from all other disjuncts. By precluding those regions to overlap, we force some examples to separate from their closest cluster. Second is the coarse partitions imposed by the tree over the variable space; a decision tree defines axis-parallel partitions that lack the flexibility of high-order polynomials, or even linear partitions with no constraint over the hyper-plane inclination. Such rigid boundaries require multiple steps before the algorithm is able to delimit a uniform-class region; a direct consequence is the undesirable fragmentation

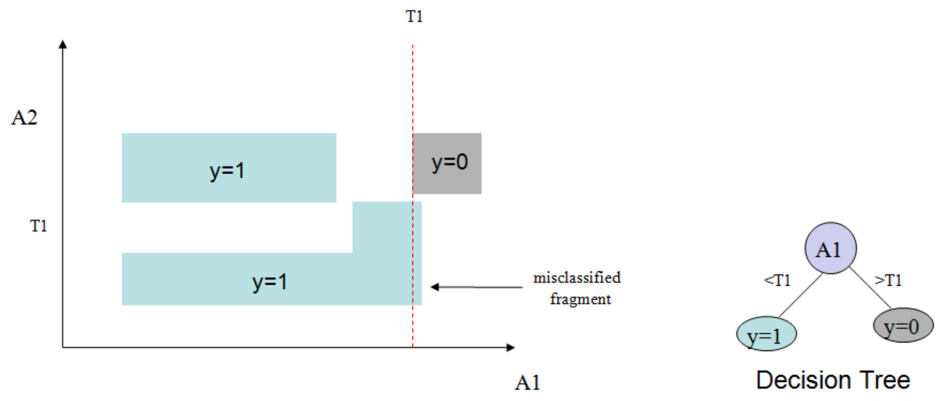

Figure 2. Left. A 2-dimensional variable space with 2 classes: signal or positive, $y = 1$, and background or negative $y = 0$. Right. The corresponding decision tree with misclassifications due to the fragmentation problem.

of class-uniform data clusters.

3. Software Tool DTFE

We now describe our software tool where the goal is to analyze the degree of fragmentation exhibited by a decision tree. The main idea is to decompose each class into clusters, where each cluster stands for a sub-concept that may suffer multiple partitions. Our tool analyzes each cluster independently and produces statistics that evaluate quantitatively the degree of fragmentation on each cluster.

3.1. Preliminary Notation

We begin with basic concepts, generalizing slightly our previous notation. Let the input space $\mathcal{X}$ be defined as before, but let the output space $\mathcal{Y} = \{y_1, y_2, \cdots, y_k\}$ comprise k different classes. Each class y_j will de decomposed into clusters $\{c_j^i\}$, where n_j is the number of clusters in y_j.

3.2. Central Mechanism

The Decision-Tree Fragmentation Evaluator is a software system that takes as input a decision tree classifier and clustered data; the output shows the corresponding fragments, clusters and classes with associated statistics. The system divides into four steps:

(i) Extract tree rules
(ii) Determine partitions over the variable space
(iii) Obtain cluster fragments
(iv) Assess cluster fragmentation

Fig. 3 shows a schematic representation of our system. As a first step we take the decision tree as input and extract the disjuncts or rule paths from the root node to each leaf node. Disjuncts are grouped by their corresponding class. Each disjunct partitions the variable space into a hyper-cube where the goal is to have a class-uniform region. At this point we are able to determine the position of each disjunct or tree rule over the variable space; this brings a clear picture of the partitioning imposed by the decision tree.

The second step takes as input a decomposition of each class into clusters. We apply a clustering algorithm to each class using the Spectral Clustering Algorithm [8]; this algorithm

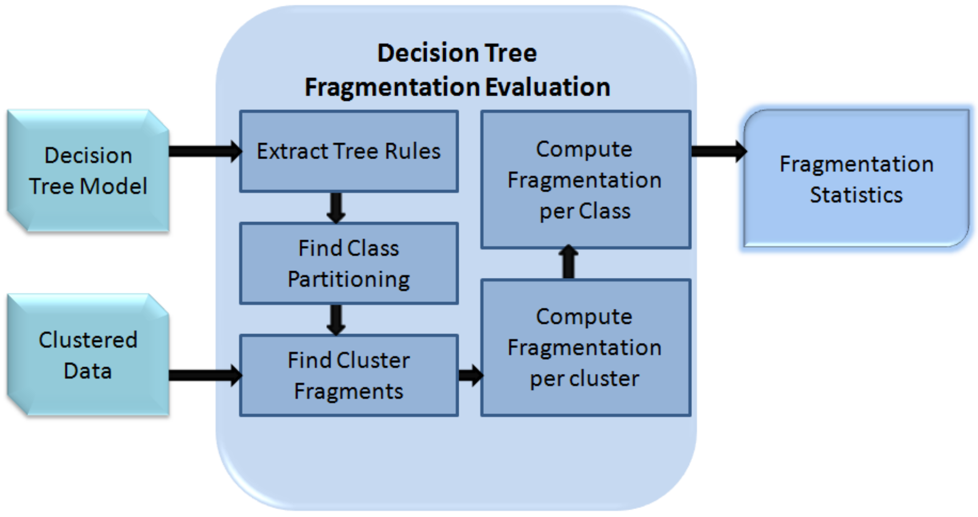

Figure 3. Architecture of DTFE, or Decision Tree Fragmentation Evaluator.

has shown excellent performance when compared to other clustering techniques, such as EM or K-means, and has the important property of being able to reduce the dimensionality of the data. Once clusters are identified, we determine how each cluster got fragmented. In essence we overlap clusters with each of the regions defined by each disjunct in the decision tree. Our goal is to capture how events from each cluster fall into the partitions induced by the decision tree.

In the next step we extract statistics related to the degree of cluster fragmentation. We compare clusters *vs* hyper-rectangles to obtain the number of fragments comprised by each cluster. For each cluster c_j^i we record the number of fragments ϕ_j^i into which the cluster got divided. We then average to obtain an estimate of the expected number of fragments per cluster on each class:

$$\Phi_j = \frac{1}{n_j} \sum_i \phi_j^i \tag{1}$$

The statistic above provides information on how much each cluster got divided on average, but disregards the proportion of events on each fragment. This is important because the presence of small fragments increases the likelihood of misclassifications. We introduce a metric for each cluster, ψ_j^i, that gives higher weight to small fragments as measured by the proportion of events on each fragment:

$$\psi_j^i = \sum_l e^{-\frac{\alpha_l}{\sigma}} \qquad 1 \leq l \leq \phi_j^i \tag{2}$$

where α_l is the proportion of events on fragment l and σ is a user-defined parameter that decides how much importance should be assigned to small proportions; in our experiments we use $\sigma = 0.1$. In essence, ψ_j^i increases when a cluster is divided into many small fragments. The value of ψ_j^i is bounded above by ϕ_j^i; it can be used for comparison purposes to determine when a cluster experiences over-fragmentation. As before, we can average to obtain an estimate of the expected value of ψ per cluster on each class:

$$\Psi_j = \frac{1}{n_j} \sum_i \psi_j^i \tag{3}$$

The statistics above form part of the output of DTFE. Such output is instrumental in assessing the quality of a decision tree as a function of the degree of fragmentation imposed on all class-clusters. The bottom-up architecture enables us to evaluate the degree of fragmentation first at the cluster level and then at the class level. We expect this will lead to a better understanding of the behavior of decision trees in real-world applications.

4. Experiments on Single Top Quark Production

Our domain lies on the search for single top quark production, a challenging problem in particle physics due to large backgrounds [9, 10]. In essence, digital signals recorded in a particle detector after each collision are used to identify certain particles. Each collision or event is characterized by variables deemed relevant to separate the desired signal, in our case single top quark, from the background. Variables are divided into three main groups: individual object kinematics, global event kinematics and angular correlations. Examples of variables include transverse momentum, invariant mass of all objects, angular separation between the two leading jets, etc. The problem is inherently challenging because single top quarks are kinematically and topologically very similar to other events, such as W+jets and $t\bar{t}$ events. Our data comes from a Monte Carlo event simulator based on the DØ detector at Fermi Lab.

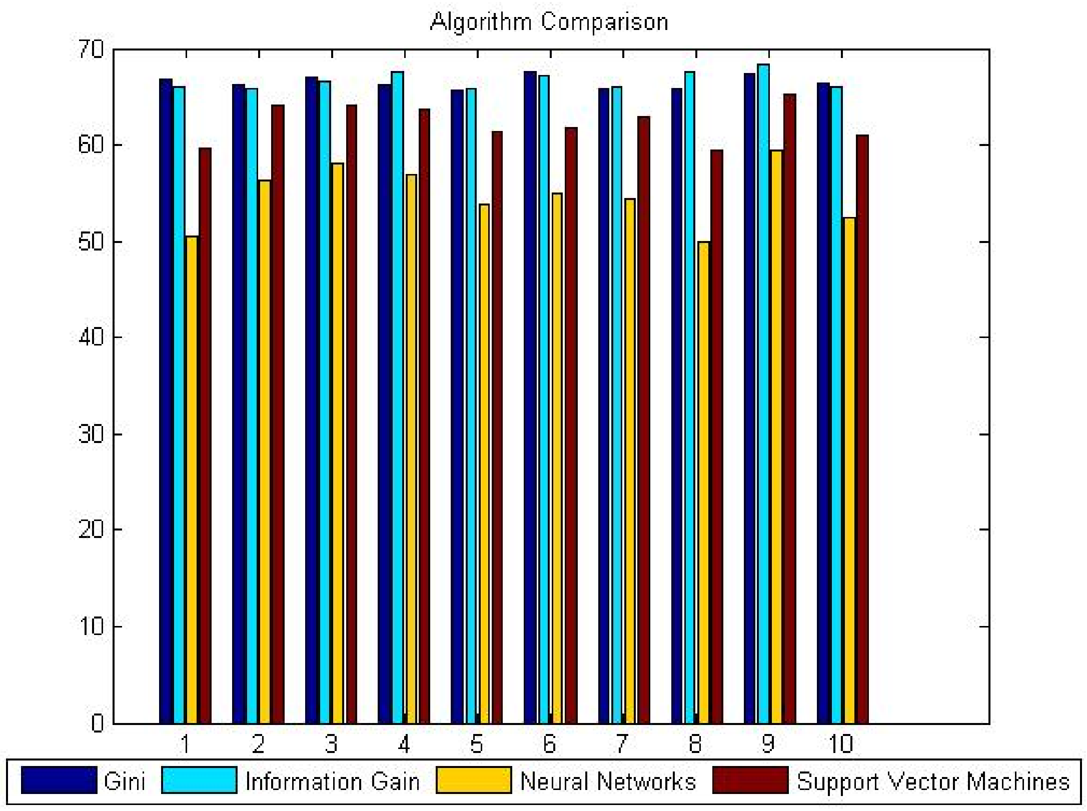

Figure 4. A comparison of DTs, NNs and SVMs under accuracy performance.

Our first experiment simply compared the performance of decision trees with other non-parametric techniques such as neural networks and support vector machines. We used accuracy as the performance metric, defined as the percentage of correctly classified examples on a testing set. We used software tool WEKA to run the classifiers.

We used two versions of decision trees corresponding to the purity metric used to select the best variable at each node: gini and information gain. The neural network was run using backpropagation; the support vector machines algorithm used a polynomial kernel of degree

Class	Φ_j (Gini)	Ψ_j (Gini)	Φ_j (Info. Gain)	Ψ_j (Info. Gain)
0td	21.4 (25.50)	11.12 (6.28)	26.4 (11.53)	17.61 (4.13)
0tl	20.85 (12.18)	15.04 (7.99)	25.09 (12.53)	17.86 (6.87)
0wb	16.89 (2.96)	11.7 (1.80)	25.93 (13.50)	14.93 (4.86)
0wc	5.61 (2.93)	2.52 (0.83)	7.54 (0.89)	3.33 (0.15)
0wl	1.47 (0.03)	0.03 (0.00)	1.32 (0.00)	0.16 (0.00)
1s	0.98 (0.02)	2.79 (0.04)	3.46 (0.11)	1.3 (0.04)
1t	2 (0.00)	0.07 (0.00)	2 (0.00)	0.47 (0.00)

Table 1. Estimated values for Φ_j and Ψ_j. Values are shown for decision trees using gini and info. gain as purity metrics. Numbers enclosed in parentheses represent standard deviations.

two. Figure 4 shows results for our ten experiments; each experiment uses as a training set a bootstrap sample from the original data. We observe that decision trees clearly outperform the other two learning algorithms; the differences between means are significant at the $p = 0.05$ level under a t-student distribution; within decision trees, impurity metric gini performs similar to information gain – differences between means are not significant.

4.1. The Output of DTFE using Top Quark Data

Table 1 summarizes the output by DTFE. The first column corresponds to the concept class, where values starting with 0 correspond to background, and those starting with 1 correspond to signal. The second and third columns show the values of Φ_j and Ψ_j when the decision tree uses impurity metric gini; the fourth and fifth columns are equivalent to the second and third columns except we use information gain as the impurity metric.

Table 1 shows results that can be used to derive conclusions about the quality of decision tees when these are generated under different parameters. In our particular domain, we observe metric gini yields lower fragments per cluster on average for every class (compare Φ_j on second and fourth columns). Statistic Ψ_j reveals more about the nature of the partitions imposed by the decision tree; one can see that info. gain tends to produce small fragments. For example, on class 1t both purity metrics produce the same value of $\Phi_j = 2$ but decision trees using info. gain produce a higher value for Ψ_j, which points to the existence of small fragments. This same effect is observed on almost all classes except for class 1s where the opposite is observed: decision trees using gini tend to produce small fragments. This analysis is useful to produce reliability estimates during prediction; according to the class under prediction, one may be better off switching from one type of decision tree to another by favoring those trees with small values for Φ_j, which points to a small number of fragments, and with large values for Ψ_j, which points to fragments that are not too small.

5. Related Work

Previous studies show how the fragmentation problem is expected to appear even outside the realm of decision tree learning, such as rule-based learning. The source of the problem can be found not only in the classification algorithm itself, but in the inherent difficulty of separating signal and background according to the data distribution. The degree of difficulty during classification can be quantified through different metrics, such as concept variation [1]; concepts with high variation are more prone to experience the effects of the fragmentation problem. A solution is to avoid the locality effect that arises when class-uniform regions in the variable space are small; this can be done by finding rules globally, looking at all training examples when generating every rule. The idea of building rules globally, by looking at all examples together, is generally accepted as a solution against data fragmentation [2].

Other studies have analyzed additional effects that tend to accompany the fragmentation problem such as tree replication and repetition [11]. All these problems are in fact related and can be tackled through feature or variable construction, where new variables are generated from the pool of original variables through logical and mathematical operators. It is interesting to observe that internal nodes in a neural network act indeed as new variables that are obtained through a linear combination of the original variables using a sigmoid function. Neural networks incorporate then some of the notions behind feature construction. If the internal nodes of a neural network are used as new variables, the resulting decision tree is less prone to fall into data fragmentation [6]. Another approach to tackle the fragmentation problem is to group variable values together before the decision tree is actually built [5]. Compared to previous work, our study introduces a quantification of the degree of fragmentation when a decision tree is under construction, which can help further understand the differences between local and global learning.

6. Summary and Conclusions

This paper describes a system called DTFE, Decision Tree Fragmentation Evaluator, that measures the degree of fragmentation exerted by a decision tree classifier on a particular dataset. The main idea is to decompose each class into clusters, and to over-impose clusters with the hyper-rectangles used by the decision tree to partition the variable space into class-uniform regions. The DTFE system outputs a set of statistics that provide useful information on the quality of the decision tree model.

Our goal is to understand the behavior of a decision tree classifier under different input-output distributions. It is important to determine when a classifier is unable to improve on accuracy, either because it is close to Bayes error, in which case no better performance is attainable, or because the bias imposed by the classifier is high, and the best chosen model is far from the true concept. Additionally the complexity of the classifier may introduce high variance to the overall error.

In future work we will try to establish a clear connection between the degree of fragmentation and the performance of a decision tree in terms of several metrics such as accuracy, precision, recall, area under the ROC curve, etc. Our goal is to contribute to the field of meta-learning where the goal is the understanding of the interaction between the mechanism of learning and the concrete contexts in which that mechanism is applicable [12]. Briefly stated, the field of meta-learning is focused on the relation between tasks or domains and learning strategies. The idea is to go beyond the goal of producing more accurate learners to the additional goal of understanding the conditions, or types of example distributions, under which a learning strategy is most appropriate.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grants no. IIS-0812372 and IIS-448542.

References

[1] Vilalta R, Blix G and Rendell L 1997 Global data analysis and the fragmentation problem in decision tree induction *Proceedings of the 9th European Conference on Machine Learning ECML* (Heinderberg: Springer-Verlag) p 312

[2] Li J and Wong L 2002 Solving the fragmentation problem of decision trees by discovering boundary emerging patterns *Proceedings of the 2002 IEEE International Conference on Data Mining ICDM* (Washington DC, USA: IEEE Computer Society) p 653

[3] Liu B, Hu M and Hsu W 2000 Intuitive representation of decision trees using general rules and exceptions *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence* (AAAI Press; MIT Press) p 615

[4] Hand D J 2006 *Statistical Science* **21** 1
[5] KMHo and Scott P 1998 Overcoming fragmentation in decision trees through attribute value grouping *Proceedings of the Second European Symposium on Principles of Data Mining and Knowledge Discovery* (Heidelberg: Springer-Verlag) p 337
[6] Liu H and Setiono R 1998 Feature transformation and multivariate decision tree induction *Proceedings of the First International Conference on Discovery Science (DS'98)* (Heidelberg: Springer Verlag) p 279
[7] DeLisle R K and Dixon S L 2004 *Journal of Chemical Information and Modeling* **44** 862
[8] Bach F R and Jordan M I 2004 Learning spectral clustering *Advances in Neural Information Processing Systems 16 NIPS* (MIT Press)
[9] Abasov V M 2006 *Physical Review D* **75**
[10] Abasov V M 2007 *Physical Review Letters* **98**
[11] Setiono R and Liu H 1998 Fragmentation problem and automated feature construction *Proceedings of the Tenth IEEE International Conference on Tools with Artificial Intelligence* (Taipei, Taiwan: IEEE Computer Society) p 208
[12] Brazdil P, Giraud-Carrier C, Soares C and Vilalta R 2009 *Metalearning: Applications to Data Mining* (Springer)